Apéndice C. Manual de usuario.

El software ha sido diseñado para ser utilizado en conjunto con las siguientes herramientas:

- CodeBlocks (Ambiente de desarrollo).
- MinGW (compilador GCC y herramientas de programación para C++).
- SDL (librería para desarrollo de aplicaciones multiplataforma).
- GLEW (librería para la activación de extenciones de OpenGL).
- T.Ed (editor de escenas en exteriores).

A excepción del editor T.ED, todas las herramientas son de libre distribución y se pueden descargar gratuitamente.

Instalación del paquete MinGW.

Básicamente lo requerimos para poder utilizar el compilador GCC. El archivo de instalación se puede descargar en la siguiente dirección: http://www.mingw.org/ .

La instalación es sumamente sencilla. Se recomienda elegir la instalación completa.

Instalación de la librería SDL.

Existe una distribución para Windows en forma de un archivo DLL. Este archivo se puede bajar en la siguiente dirección: http://www.libsdl.org . El archivo DLL debe ser copiado dentro de la carpeta BIN del compilador cuya hubicación generalmente es la siguiente: C:\MinGW\bin

Instalación de la librería GLEW.

Existe una distribución para Windows en forma de un archivo DLL. Este archivo se puede bajar en la siguiente dirección: http://glew.sourceforge.net/ . El archivo DLL

debe ser copiado dentro de la carpeta BIN del compilador cuya hubicación generalmente es la siguiente: C:\MinGW\bin

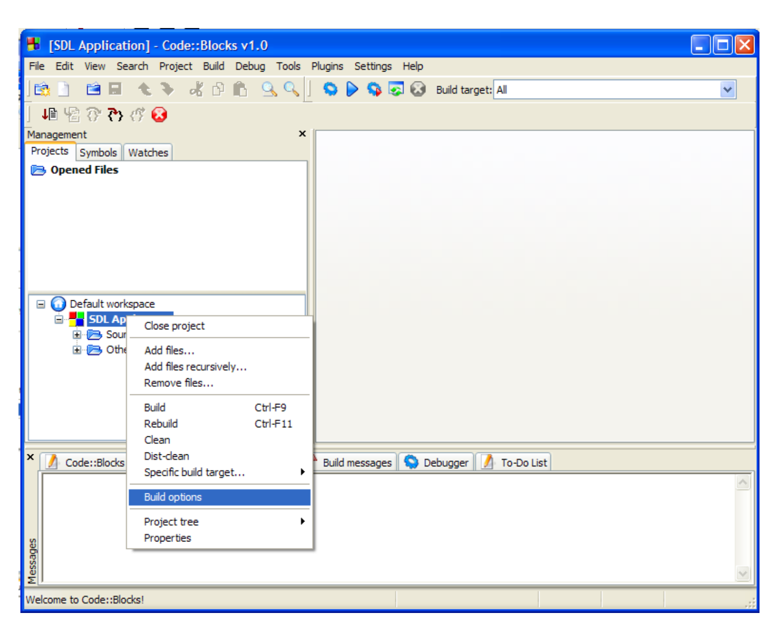
Instalación del software desarrollado.

El software se distribuye como un archivo DLL y debe ser copiado dentro de la carpeta BIN del compilador cuya hubicación generalmente es la siguiente: C:\MinGW\bin

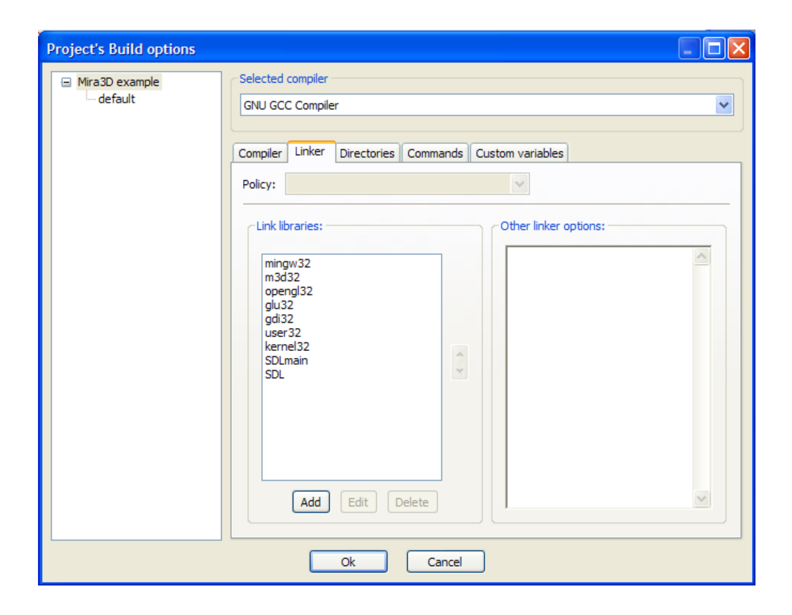
Instalación y configuración del ambiente de desarrollo CodeBlocks.

El software se puede bajar en la siguiente dirección: http://www.codeblocks.org/ . Opcionalmente se puede bajar junto con el compilador MinGW pero no se recomienda. La primera vez que se ejecuta el programa se presenta una pantalla con los compiladores disponibles. Elegir “GCC”.

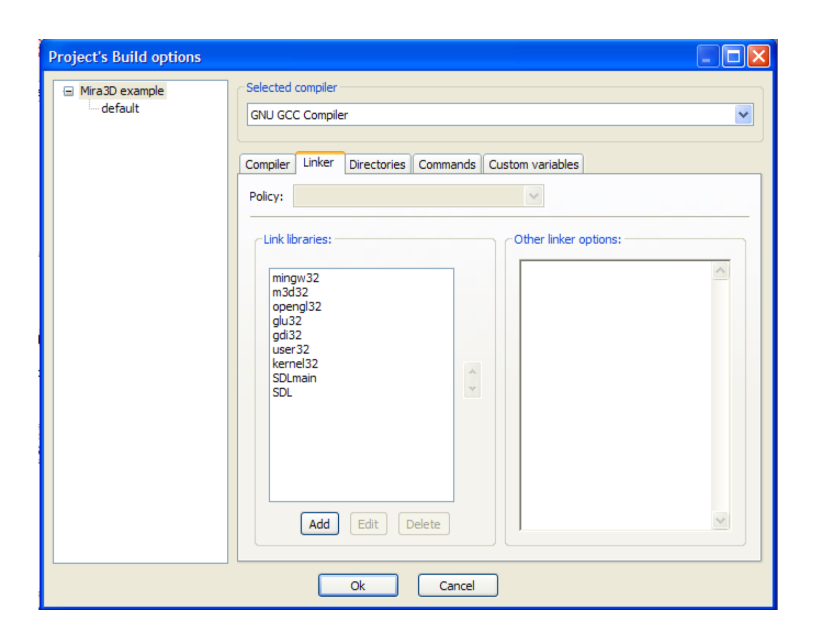
Una vez instalado se debe proceder a su correcta configuración para poder ser utilizado con el software desarollado. Primero que nada es necesario crear un nuevo proyecto cuya opción esta localizada en el menú “File”, y elegimos crear un nuevo proyecto de “SDL”. Una vez hecho lo anterior, damos clic derecho al nombre del proyecto creado y elegimos la opción “build options” como se muestra en la siguiente figura:



En la siguiente pantalla elegimos la pestaña “linker” y las siguientes variables deben estar registradas. Es importante respetar el orden en que aparecen:



Una vez hecho este presionamos el boton OK para registrar los cambios. Ahora ingresamos al menú “Project” y seleccionamos la opcion “Build options”. Aparecerá otra ventana similar a la de la figura anterior debemos ingresar las mismas variables que en la ocasión anterior, cuidando igualmente el orden en que deben aparecer registradas:



Finalmente debemos de copiar los archivos de encabezado del software desarrollado dentro de la carpeta “include” del compilador (generalmente se localiza en la siguiente ruta: C:\MinGW\include). Ahora estamos listos para escribir un programa.

Tutorial para el uso del software.

Primero es necesario crear un archivo llamado “init.ini” que debera localizarse en la ruta en donde se encuentra el archivo ejecutable del programa que será creado. Este archivo contiene todas las opciones que requerimos para la inicialización de un programa. A continuación se muestra un ejemplo de archivo “init.ini” con los par´metros recomendados:

```
### Informacion sobre la pantalla ###

width 1024
height 768
colordepth 32
fullscreen 1
windowtitle M3D App

### Informacion sobre la camara ###

eyepos 0.0 0.0 0.0
lookat 0.0 0.0 1.0
upvector 0.0 1.0 0.0

viewangle 45.0
aspectratio 1.6
neardist 0.1
fardist 1200.0
midpercdist 0.65
fogcolor 192 192 192

### Opciones avanzadas ( Se recomienda dejar estos valores ) ###

slidespeed 2
rotspeed 0.11
slerpincrement 0.05
```

```
### Imagenes de la barra de progreso ( los sprites deben ser cuadrados para verse sin deformacion ) ###

spritesidelength 0.2
spritepath media/sprites/c1.bmp
spritepath media/sprites/c2.bmp
spritepath media/sprites/c3.bmp
spritepath media/sprites/c4.bmp
spritepath media/sprites/c5.bmp
spritepath media/sprites/c6.bmp
spritepath media/sprites/c7.bmp
spritepath media/sprites/c8.bmp
spritepath media/sprites/c9.bmp
spritepath media/sprites/c10.bmp
spritepath media/sprites/c11.bmp
spritepath media/sprites/c12.bmp
spritepath media/sprites/c13.bmp
spritepath media/sprites/c14.bmp
spritepath media/sprites/c15.bmp
endsprite
```

A continuación daremos una descripción de cada una de estas opciones:

width, *height* : Resolución de pantalla

colordepth : Profundidad de color de pantalla.

fullscreen : 1 para usar pantalla completa o 0 para utilizar modo ventana.

windowtitle : Texto que aparecerá en la ventana (si se elije dicho modo)

eyepos : Posición inicial de la cámara en el mundo 3D.

lookat : Orientación inicial de la cámara.

upvector : Dirección del vector UP. Por lo genera siempre deberia ser 0 1 0 .

viewangle : Angulo de amplitud de la cámara.

aspectratio : Generalmente es 1.6 para monitores full screen y 1 para monitores normales.

Neardist : Distancia a la cámara del plano "near" del view frustum.

fardist : Distancia a la cámara del plano "far" del view frustum.

midpercdist : Distancia a la cámara del plano intermedio. Se specifica como un porcentaje respecto a la distancia del plano "far" del view frustum, es decir, si

especificamos por ejemplo un valor de 0.5 significa que este plano se localizara a la mitad de la profundidad a la cual se localiza el plano “far”.
fogcolor : Valores RGB del color que deseamos tenga el efecto de niebla.

Slidespeed : Veocidad de traslación de la cámara.
rotspeed : Velocidad de rotación de la cámara.
slerpincrement : Constante para el control del movimiento de inercia de la cámara.

spritesidelength : Especifica el tamaño que tendrá el cuadro en donde será desplegada la animación de la barra de progreso.
spritepath : Nombre y ruta de cada cuadro de animación.
endsprite : Fin de la secuencia de animación.

La estructura básica de un programa con todas las funciones incluidas es la siguiente:

```
#include <M3D/M3D_engine.h>

int main( int argc, char* argv[] )
{
   M3DUser initobj;

   initobj.drawOctants( false, false );

   initobj.addTiledTerrain( "media/epr/epr2/terrain.txt", 3, 8, M3D_AUTO_MODE );

   initobj.skyBox( "media/skybox/front.bmp", "media/skybox/back.bmp",
                   "media/skybox/right.bmp", "media/skybox/left.bmp",
                    "media/skybox/up.bmp", "media/skybox/down.bmp" );

   initobj.beginOctree( 3, 8, M3D_AUTO_MODE );
       initobj.addMesh( "media/props/dwarf2.obj", true, 1.0f, 1.0f, 1.0f,
                                                        0.0f, 0.0f, 0.0f,
                                                        0.0f, 0.0f, 0.0f );
       initobj.addStaticProps( "media/epr/epr2/castle.epr" );
   initobj.endOctree();

   initobj.runEngine();

   return 0;
}
```

En el programa primero se incluye el archivo de encabezado M3D_engine para poder utilizar el software. Posteriormente dentro del programa principal se crea un objeto de tipo M3DUser que contiene los métodos que pueden ser utilizados.

A continuación se describen los métodos utilizados.

addMesh(char* *filename*, bool *startCentered*,
M3DFloat *scaleX*, M3DFloat *scaleY*, M3DFloat *scaleZ*,
M3DFloat *transX*, M3DFloat *transY*, M3DFloat *transZ*,
M3DFloat *rotX*, M3DFloat *rotY*, M3DFloat *rotZ*)

Este método sirva para insertar directamente un modelo 3D dentro de la escena.

filename : Indica la ruta y nombre del archivo el cual debe tener formato '.obj'.

startCentered : Indica si queremos que el modelo se ubique inicialmente en el centro de la escena y posteriormente podemos aplicarle transformaciones.

scaleX : Escalamiento sobre el eje X (1.0 = 100% del tamaño).

scaleY : Escalamiento sobre el eje Y (1.0 = 100% del tamaño).

scaleZ : Escalamiento sobre el eje Z (1.0 = 100% del tamaño).

transX : Translación sobre el eje X.

transY : Translación sobre el eje Y.

transZ : Translación sobre el eje Z.

rotX : Rotación en el eje X (en grados).

rotY : Rotación en el eje Y (en grados).

rotZ : Rotación en el eje Z (en grados).

return : True o False si se añadió o no con éxito la información.

addStaticProps(char* *filename*)

Este método se utiliza para importar un archivo EPR del editor T.ED. En este archivo se encuentran definidos los objetos de una escena. Es importante señalar que todos los archivos 3D y de textura a los que se haga referencia, deben estar localizados en la misma ruta en donde se encuentra este archivo EPR. Ademas, solo se aceptan archivos 3D de formato OBJ.

filename : Nombre y ruta del archivo EPR-

addTiledTerrain(char* *filename*, M3DUint *displayLevel*,
M3DUint *collisionLevel*, draw_mode *drawMode*)

Sirve para añadir un terreno seccionado por el editor T.ED. Es requerido convertir los archivos 3D generados por este editor al formato OBJ ya que solo este formato es compatible. El terreno es almacenado en un octree utilizado unicamente para el terreno.

filename : Nombre y archivo del archivo TXT que contiene los nombres y rutas de todos los archvivos 3D que confirman el terreno. Todos los archivos que componen el terreno, tanto 3D como de textura, deben estar en la misma ruta que este archivo TXT.

displayLevel : Nivel de profundidad en el octree en donde se localizarán los octantes usados para almacenar la información geométrica de los objetos 3D. El nivel 0 corresponde al nodo padre.

collisionLevel : Nivel de profundidad en el octree en donde se localizarán los octantes usados para realizar la detección de colisiones con la geometría que delimita el octree. El nivel 0 corresponde al nodo padre.

Un ejemplo de archivo TXT es el siguiente:

```
w 500
h 500
t terr(1,1).obj
t terr(2,1).obj
t terr(1,2).obj
t terr(2,2).obj
```

En donde:

w : Ancho del terreno.
h : Largo del terreno.
t : Despues de esta palabra reservada debe escribirse el nombre de archivo para cada sección del terreno.

beginOctree(M3DUint *displayLevel*, M3DUint *collisionLevel*,
draw_mode *drawMode*)

Sirve para crear un Nuevo octree dentro del cual pueda ser almacenados los objetos 3D registrados por el usuario.

displayLevel : Nivel de profundidad en el octree en donde se localizarán los octantes usados para almacenar la información geométrica de los objetos 3D. El nivel 0 corresponde al nodo padre.

collisionLevel : Nivel de profundidad en el octree en donde se localizarán los octantes usados para realizar la detección de colisiones con la geometría que delimita el octree. El nivel 0 corresponde al nodo padre.

drawMode : Puede elegirse una de las siguientes cinco opciones: M3D_PLAIN_DIRECT_MODE, M3D_PLAIN_VERTEX_ARRAY_MODE, 3D_PLAIN_VERTEX_BUFFER_MODE, _DISPLAY_LIST_MODE, M3D_AUTO_MODE. La opción M3D_AUTO_MODE es la más recomendada ya que elige el modo más adecuado automáticamente.

Todos la información 3D añadida abajo de este método serán añadidos a este octree hasta que se ejecute el método *endOctree()* que indicara su cierre.

drawOctants(bool *drawDisplayOctants*, bool *drawCollisionOctants*)

drawDisplayOctants : "true" o "false" si queremos o no dibujar los límites de los octantes utilizados para almacenar la información geométrica.

drawCollisionOctants : "true" o "false" si queremos o no dibujar los límites de los octantes utilizados para realizar la detección de colisiones.

Este método es opcional. No es necesario incluirlo.

endOctree()

Se utiliza para indicar que se ha terminado de añadir información al octree actual.

runEngine()

Se llama al final del programa para inicializar la ejecución del motor gráfico.

**skyBox(char* *frontFilename*, char* *backFilename*,
char* *rightFilename*, char* *leftFilename*,
char* *upFilename*, char* *downFilename*)**

Con este método indicamos que queremos utilizar un skybox y en sus parámetros se especifican los archivos gráficos que se usaran para cada cara del mismo. Los archivos de las texturas pueden estar en formato bmp o tga.

frontFilename : Archivo gráfico con la textura de la cara interior frontal.

backFilename : Archivo gráfico con la textura de la cara interior trasera.

rightFilename : Archivo gráfico con la textura de la cara interior derecha.

leftFilename : Archivo gráfico con la textura de la cara interior izquierda.

upFilename : Archivo gráfico con la textura de la cara interior superior.

downFilename : Archivo gráfico con la textura de la cara interior inferior.